Configurer Nexus 9000 comme générateur de trafic avec SCAPY

Table des matières

Introduction

Ce document décrit Scapy, un outil Python de manipulation de paquets pour les commutateurs N9K pour créer et manipuler des paquets avec facilité.

Conditions préalables

Téléchargez Scapy sur le bootflash du commutateur.

Pour télécharger Scapy, utilisez le lien de GitHub [GitHub-SCAPY](#)

Exigences

Cisco vous recommande de prendre connaissance des rubriques suivantes :

- Commutateur Nexus 9000/3000

Composants utilisés

- N9K-C9396PX

The information in this document was created from the devices in a specific lab environment. All of

the devices used in this document started with a cleared (default) configuration. Si votre réseau est en ligne, assurez-vous de bien comprendre l’incidence possible des commandes.

Installation

Téléchargez et extrayez le code Scapy dans la mémoire flash de démarrage de votre commutateur ; FTP, SFTP ou SCP sont disponibles.

Activez la fonction, dans ce cas, SCP.

```
switch(config)# feature scp-server
switch(config)# sh feature | i scp
scpServer              1          enabled
```

Copiez le fichier sur le commutateur depuis l'ordinateur portable.

```
scp scapy-vxlan-master.zip admin@10.88.164.13:/
```

Une fois que l'image est dans la mémoire flash de démarrage, elle doit être décompressée. Il doit activer la fonctionnalité bash et la décompresser de bash.

```
switch(config)# feature bash
switch(config)# run bash
bash-4.3$ sudo su -
root@switch#cd /bootflash
root@switch#unzip scapy-vxlan-master.zip
```

Une fois décompressés, les fichiers peuvent être localisés avec la commande dir sur la mémoire flash de démarrage, les fichiers compressés et décompressés.

```
switch# dir bootflash: | i i scapy
       4096    Jul 09 18:00:01 2019  scapy-vxlan-master/
    1134096    Jul 19 23:35:26 2023  scapy-vxlan-master.zip
```

Scapy est maintenant disponible.

Notez que vous devez appeler le programme avec des privilèges racine et que vous devez également naviguer jusqu'au répertoire Scapy.

```
switch(config)# run bash
Enter configuration commands, one per line. End with CNTL/Z.
bash-4.2$ sudo su -
root@switch#cd /
root@switch#cd bootflash/scapy-vxlan-master       <<< Move to the scapy folder scapy-vxlan-master
root@switch#python                                <<< Run python once located inside the folder
Python 2.7.2 (default, Mar  9 2015, 15:52:40)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from scapy.all import *                       <<< Import libraries from scapy
>>>
```

Créer un paquet

Ceci est un exemple de la façon de créer un paquet IP de base pour illustrer la procédure pour générer du trafic à l'aide de Scapy.

```
Create l2 source and destination mac addresses.
>>> l2=Ether()
>>> l2.src='00:aa:12:34:12:34'
>>> l2.src='00:ff:aa:bb:cc:11'

Create l3 source and destination IP addresses.
>>> l3=IP()
>>> l3.src='10.1.1.1'
>>> l3.dst='10.2.2.2'
```

Une autre fonctionnalité consiste à envoyer un paquet à partir d'un fichier pcap précédemment capturé. Pour ce faire, utilisez la commande rdpcap.

La sortie de cette commande est une liste Python contenant tous les paquets capturés dans votre fichier pcap. Dans cet exemple, traffic.pcap contient 10 paquets et ces paquets sont assignés à la liste créée en tant que paquets.

```
>>> pkts = rdpcap('bootflash/traffic.pcap')
>>> len(pkts)
```

```
10
>>> type(pkts)
<class 'scapy.plist.PacketList'>
```

Remarque : le fichier pcap doit être stocké dans la mémoire flash de démarrage du commutateur.

Envoyer le trafic

Une fois le paquet créé, nous utilisons la commande sendp pour commencer à envoyer notre paquet sur l'interface spécifiée.

```
>>> packet = l2/l3.                   << packet now have the values for source and destination declared
>>> sendp(packet, iface='Eth1-1').    << Sending the packet through interface eth1/1
.
Sent 1 packets.
```

Vous pouvez ensuite parcourir la liste de paquets pour envoyer le trafic sur l'interface que vous spécifiez.

```
>>> while True:
...  for i in range(len(pkts)):         <<< It goes through the list pkts with 10 packets and send 1 by
...   sendp(pkts[i], iface='Eth1-1')
...
.
Sent 1 packets.
.
Sent 1 packets.
```

Remarque : seuls les ports de commutateur en mode d'accès peuvent être utilisés. Sinon, il affiche une erreur.

Exemple de l'erreur :

```
>>> sendp(l2/l3, iface='Eth1-6')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "scapy/sendrecv.py", line 335, in sendp
socket = socket or conf.L2socket(iface=iface, *args, **kargs)
```

```
File "scapy/arch/linux.py", line 477, in __init__
set_promisc(self.ins, self.iface)
File "scapy/arch/linux.py", line 165, in set_promisc
mreq = struct.pack("IHH8s", get_if_index(iff), PACKET_MR_PROMISC, 0, b"")
File "scapy/arch/linux.py", line 380, in get_if_index
return int(struct.unpack("I", get_if(iff, SIOCGIFINDEX)[16:20])[0])
File "scapy/arch/common.py", line 59, in get_if
ifreq = ioctl(sck, cmd, struct.pack("16s16x", iff.encode("utf8")))
IOError: [Errno 19] No such device
```

Assurez-vous que l'interface est utilisable, exécutez la commande ifconfig, l'interface doit y être répertoriée.

```
bash-4.3$ ifconfig | grep Eth
Eth1-1 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:88
Eth1-2 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:89
Eth1-5 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:8c
Eth1-6 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:8d
Eth1-8 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:8f
Eth1-11 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:c1
...
```

Vérifier

Vous pouvez utiliser la commande pour vérifier un paquet donné.

```
>>> pkts[5].show()
###[ Ethernet ]###
  dst       = 01:00:0c:cc:cc:cd
  src=58:97:bd:00:a4:f2
  type      = 0x8100
###[ 802.1Q ]###
     prio      = 6
     id        = 0
     vlan      = 104
     type      = 0x32
###[ LLC ]###
        dsap      = 0xaa
        ssap      = 0xaa
        ctrl      = 3
###[ SNAP ]###
           OUI       = 0xc
           code      = 0x10b
###[ Spanning Tree Protocol ]###
              proto     = 0
              version   = 2
              bpdutype  = 2
```

```
              bpduflags = 60
              rootid    = 32872
              rootmac   = 58:97:bd:00:a4:f1
              pathcost  = 0
              bridgeid  = 32872
              bridgemac = 58:97:bd:00:a4:f1
              portid    = 32769
              age       = 0.0
              maxage    = 20.0
              hellotime = 2.0
              fwddelay  = 15.0
###[ Raw ]###
                 load      = '\x00\x00\x00\x00\x02\x00h'
```

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.